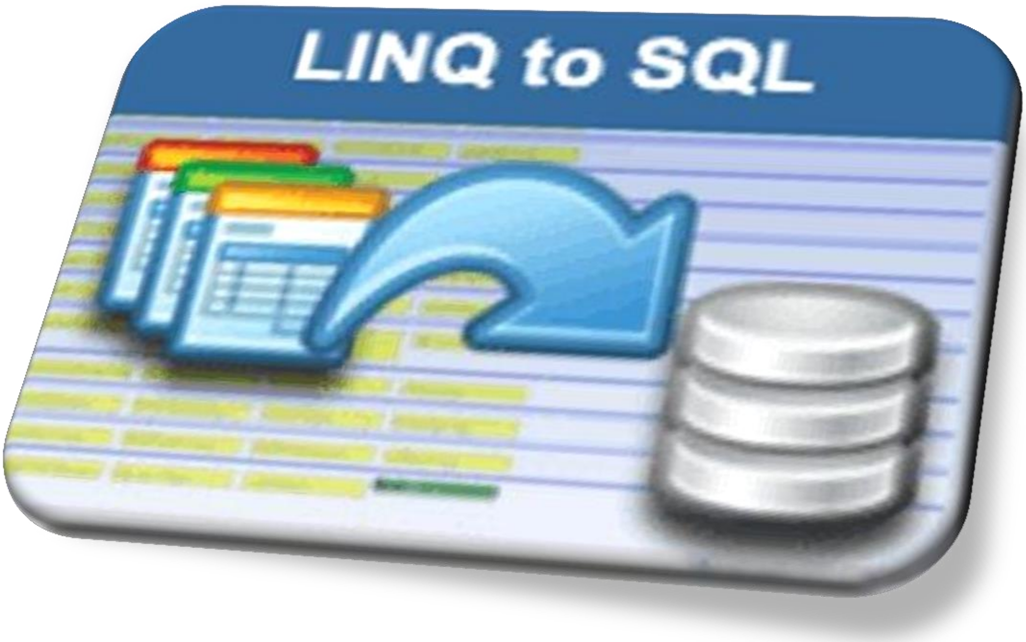
منتدى فيجوال بيسك لكل العرب

عندما تبدع انامل العرب



LINQToSQL Using C#

اعداد :

سجاد محمد باقر (sajad)

مشرف منتدى فيجوال بيسك لكل العرب vb4arb

العراق – كركوك

بسم الله الرحمن الرحيم

وصلى الله على محمد وآله الطاهرين

السلام عليكم ورحمة الله وبركاته

مقدمة:

سنتطرق في هذا الكتيب الى مقدمة بسيطة عن تقنية LINQ وبعد ذلك نبين كيفية استخدامها مع قواعد البيانات مع مثال بلغة C# وقواعد بيانات SQL Server 2008.

تعريف:

LINQ: مختصر لجملة Language Integrated Query أي ما معناه لغة الاستعلام المتكامل, وهي تقنية او نموذج ومنهج برمجي مقدم من Microsoft لتضيف قدرات الاستعلام الاساسية المستخدمة مع قواعد البيانات الى لغات البرمجة المستندة على الـNET., حيث تقدم هذه التقنية تعابير واضحة وذكية لمعالجة البيانات, وتكمن القدرة الحقيقية لهذه التقنية في قابليتها على تطبيق نفس الاستعلام على قاعدة بيانات SQL, على Dataset, مصفوفة من البيانات في الذاكرة, XML, الكائنات والعديد من الانواع الاخرى من البيانات.

الجزء المتكامل معناه أن التقنية جزء من بناء الجملة في لغة البرمجة, اما الجزء الاخر (الاستعلام) يوضح قدرة وقوة هذه التقنية من حيث معالجتها العديد من الانواع من البيانات ,ويمكن وصف هذه التقنية أيضا بأنها تعابير او جمل برمجية يستخدم للاستعلام عن البيانات.

فوائدها:

1- سهولة التعامل مع البيانات.
2- تحويل الجداول الى كائنات والاعمدة الى خصائص وانشاء العلاقات تلقائيا اثناء التحويل.
3- الاستعلام عن العديد من الانواع من البيانات, حيث لا يقتصرعلى قواعد البيانات العلائقية.
4- استخدام تعابير لامبدا.
5- استخدام LINQ مع قواعد البيانات هي أكثر أمانا من تقنية ADO.NET.
6- أكوادها قصيرة.

عيوبها:

1- أن تغيير طريقة الوصول الى البيانات يحتاج منك اعادة الترجمة.
2- من الصعب فهم الاستعلام من خلال التعابير المعقدة.
3- عملية الربط (Joins) تتسم بالبطئ.
4- عدم وجود مخطط واضح للطبقات (Tiers).
5- ارسال الاستعلام بأكمله الى قاعدة البيانات وهذا يؤدي الى أخذ الكثير من حركة المرور في الشبكة.

انواع تقنية LINQ:

1- DLINQ: استخدام تقنية LINQ مع قواعد بيانات SQL. (وهذا ما سنشرحه في هذا الكتيب ان شاءالله).
2- XLINQ: استخدام تقنية LINQ مع ملفات XML.
3- LINQ: استخدام تقنية LINQ مع الكائنات (Objects).

مقارنة بين استعلام LINQ واستعلام SQL

اولا: صيغة الاستعلام الخاص بتقنية LINQ:

```
IEnumerable<T> or Anonymous (var) = from <<element>> in <<Source collection>>
   where <<Condition As Boolean Output>>
   select <<expression>>
```

مثال:

```
var items = from name in names where name == "أحمد" select name;
```

ان ناتج الاستعلام يكون من نوع (generic collection) بشكل عام ومن الممكن أن يكون الناتج قيم مجهولة حيث يتم التعرف على نوع الارجاع من قبل المترجم ففي هذه الحالة لايجب استخدام الفئة <IEnumerable<T كناتج للاستعلام, أما نوع الارجاع <T> يحدده العبارة Select بمعنى هل الارجاع من نوع عدد صحيح, سلسلة نصية, فئة, والخ. حسب شرط معين ويمكن اهمال الشرط في حال عدم الحاجة اليه, اما <Source Collection> فهي مصدر البيانات المراد الاستعلام منها.

ثانيا: صيغة الاستعلام الخاص لـSQL المشابه لاستعلام LINQ أعلاه فهي:

```
Select <Columns> from <table> where <Conditions>
```

لاحظ التشابه الكبير بين صيغتي الاستعلامين الفرق فقط في التقديم والتأخير بين الكلمات المحجوزة مثل كلمات (from ,select ......etc).

نفس المثال اعلاه لكن في SQL

```
Select name from table where name = 'محمد'
```

سنأخذ مثالا بسيطا لنبين كيفية استعمال تقنية LINQ:

لو كانت لدينا مصفوفة من الاسماء

```
string[] names = new string[] { "سجاد", "أحمد", "محمد", "أمير", "سهى",
"عمار", "جمال" };
```

وطلب منا البحث عن اسم معين فيكون الحل كالتالي:

```
for (int i = 0; i < names.Length; i++)
            {
                if(names[i] == "محمد")
                {
                    MessageBox.Show("Found");
                }
            }
```

هذا طبعا بدون استخدام LINQ, أما لو بحثنا عن الاسم باستخدام LINQ فسيكون البحث بهذه الطريقة:

الحل:

```
            IEnumerable<string> namecollection = from name in names where name ==
"محمد" select name;

            foreach (string name in namecollection)
            {
                MessageBox.Show(name);
            }
```

في المثال اعلاه تم استخدام الفئة IEnumerable<T> كناتج للاستعلام وهذا طبيعي لان مصدر البيانات عبارة عن سلسلة نصية فليس هنالك من داع لاستخدام متغير من نوع var لكن ماذا لو كانت مصدر البيانات عبارة عن قيم غير معروفة النوع؟ في هذه الحالة يسمى ناتج الاستعلام بالناتج المجهول لانه لايمكن التكهن بناتج الاستعلام وهنا تأتي دور الكلمة المحجوزة var.

var: عبارة عن نوع من الانواع الضمنية حيث يقوم بتمثيل أي نوع من البيانات ويحدد نوع البيانات كما قبل المترجم.

لنأخذ مثالا يبين ذلك:

لدينا الفئة التالية باسم Books:

```
class Books
        {
            public int BookID { set; get; }
            public string BookName { set; get; }
        }
```

المطلوب أن يكون ناتج الاستعلام البحث عن كتاب معين حسب التسلسل ومطلوب ايضا تغيير اسم الخاصية BookName الى اسم آخر مثلا BName.

ملاحظة: تتيح لك تقنية LINQ تغيير اسماء الخصائص اثناء الاستعلام باستخدام الكلمة المحجوزة new باسماء اخرى غير معروفة النوع, بمعنى آخر يمكن أختيار حقول دون أخرى.

الحل:

```
Books[] books = new Books[] {
            new Books{ BookID = 1, BookName = "C#"},
            new Books{ BookID = 2, BookName = "VB"}
            };

            var book = from b in books where b.BookID == 1 select new { BName = b.BookName };

            foreach (var item in book)
            {
                MessageBox.Show(item.BName);
            }
```

لاحظ هذا السطر: `select new { BName = b.BookName }`

تم تغيير اسم الكتاب الى اسم أخر غير معروف النوع وذلك باستخدام الكلمة المحجوز new ففي هذه الحالة يجب تعريف متغير من نوع var كناتج للاستعلام وايضا داخل الـForEach ويتم عرض الاسم عن طريق الاسم الجديد BName.

المزيد من استعلامات LINQ:

```
// Display all books order by book name
var book = from b in books orderby b.BookName select b;
// Display all books order by book id but this time descending
var book = from b in books orderby b.BookID descending select b;
// Using Lambda Expression and Function OrderBy
var book = books.OrderBy(c => c.BookName).Select(c => c);
// Retrieving books according to BookID and Renaming the Source Name of properties
var book = from b in books where b.BookID == 1 select new { ID = b.BookID, BName = b.BookName
};
//
```

```
int[] numbers = new int[] { 2, 4, 1, 3, 6, 7, 5, 8, 9 };
// Display odd numbers
var odds = from num in numbers where num % 2 != 0 select num;
// Retrieving Names witch start with 'a' letter
var name = from n in names where n.Name.StartsWith("a") select n;
// Displaying Single Book using Lambda Expression depending on book id
Books book = books.Single(c => (c.BookID == 1));
// Displaying book name where book id equal to 2 (output as IEnumerable<string> or var)
var book = books.Where(bid => bid.BookID == 2).Select(bn => bn.BookName);
// Returning number books in collection
int bcount = books.Count();
// you can retrieving data from two collection as showing below
ba book = from b in books
                       from a in aouthers
                       select new { b.BookName, a.aname };
// You can join multiple collection using join keyword as showing below
var bajoin = from b in books
                         join a in aouthers on b.aid equals a.aid
                         select new { a.aid, b.BookName, a.aname };
```

بناء مشروع يبين كيفية التعامل مع قواعد البيانات SQL باستخدام تقنية LINQ:

سنقوم ببناء مشروع يتعامل مع تقنية LINQ مع قواعد بيانات SQL يؤدي العمليات الاساسية التي تحتاجها كل برنامج قاعدة بيانات من (اضافة, حذف, تعديل, بحث, طباعة تقرير سنستخدم تقارير مايكروسوفت (Report.rdlc)) واستعراض السجل السابق واللاحق) لكي يكون مرجعا لكل من يريد تعلم هذه التقنية واستخدامها مع قواعد بيانات SQL.

بسم الله نبدأ

اولا : قم بإنشاء قاعدة بيانات باسم std عن طريق برنامج ال SQL Management Studio ومن ثم أنشئ جدول باسم StdInfo
وأضف اليه الحقول (ID, FName, DoB, Adress) واضبط خصائص الحقول كما في الشكل الآتي:



مع جعل حقل ال IDمفتاح اساسي و ترقيم تلقائي من خاصية Identity column كما هو موضح في الصورة اعلاه.

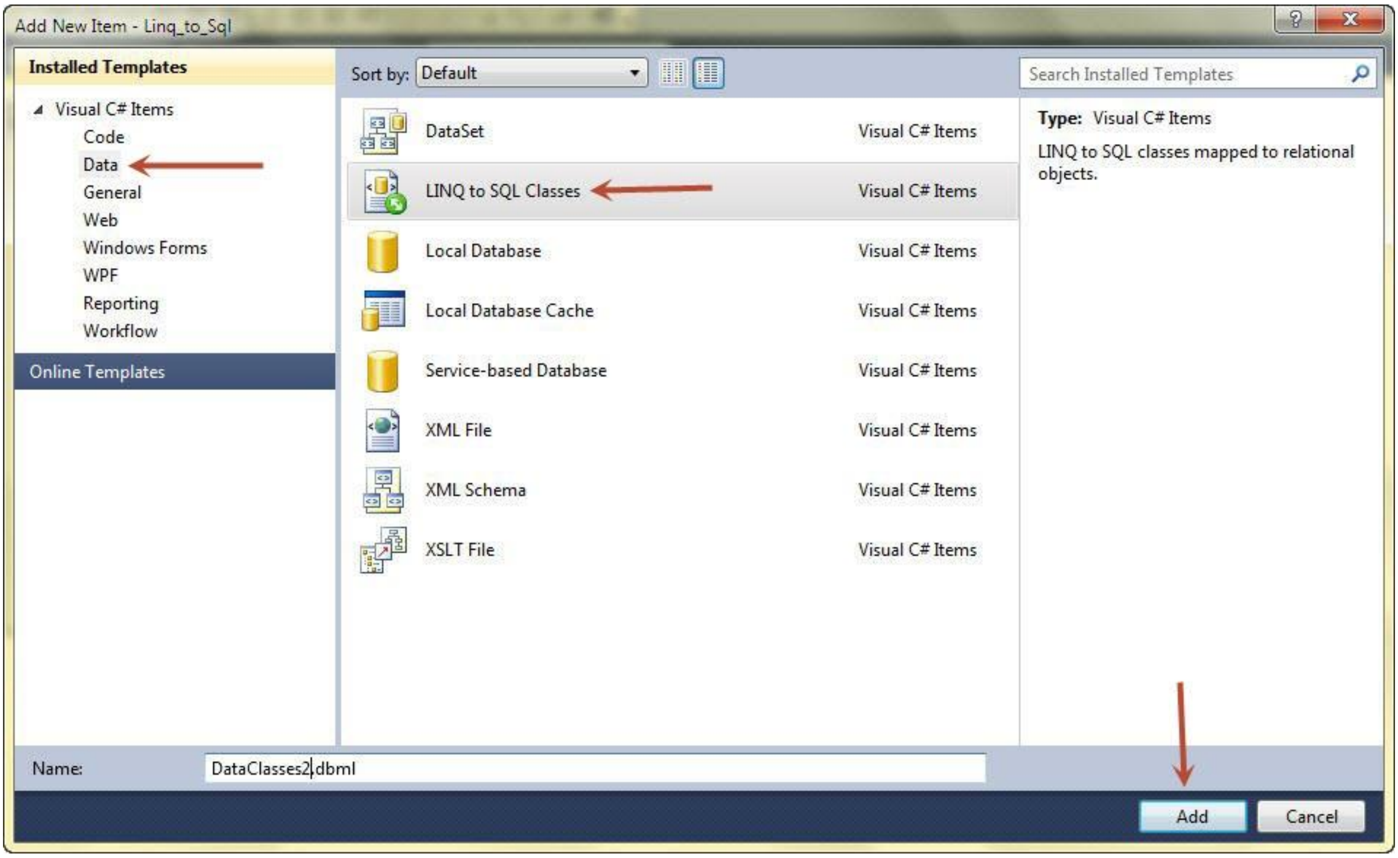
ثانيا : افتح مشروع جديد باسم Lint_to_SQL وصمم الواجهة بالشكل الآتي:



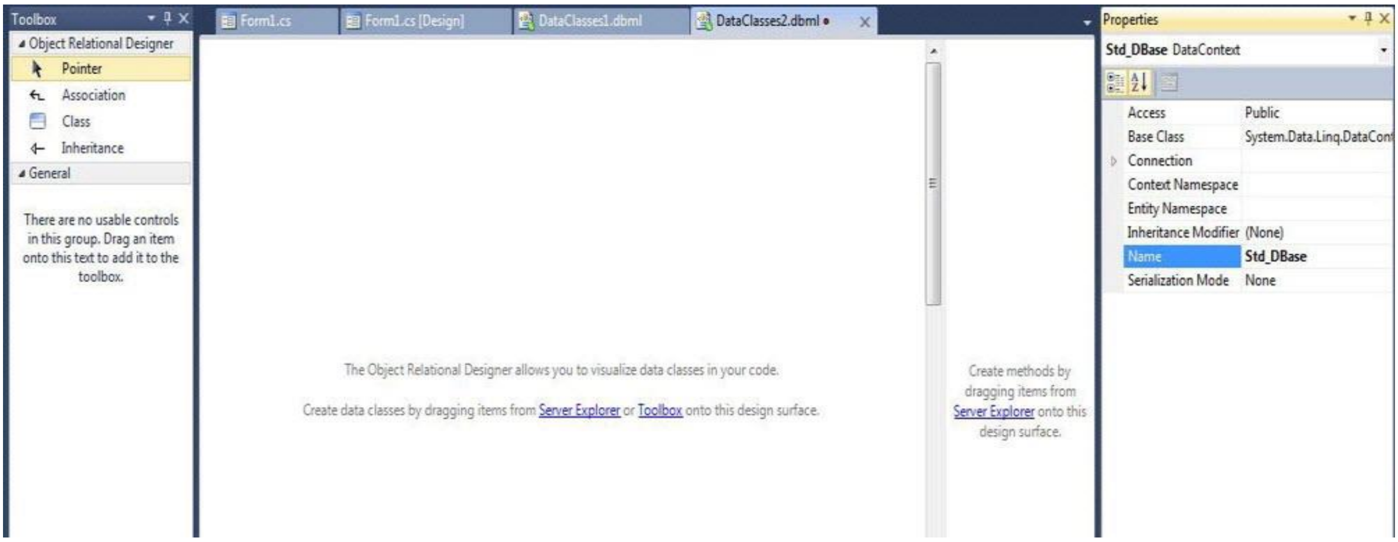
قم بضبط خصائص الادوات كما في الجدول الآتي:

الاداة	الاسم	الاستخدام	خصائص اضافية
TextBox	Id_txt	تسلسل الطالب	RightToLeft=Yes Enabled=False
TextBox	name_txt	اسم الطالب	
TextBox	address_txt	عنوان الطالب	
MaskedTextBox	dob_txt	تاريخ الولادة	RightToLeft=No TextAlign=Right
TextBox	srch_txt	حقل البحث بالاسم	RightToLeft=Yes AutoCompleteMode= SuggestAppend AutoCompleteSpurce= CustomSource
button	new_btn	تفعيل اضافة طالب جديد	
button	cancel_btn	الغاء تفعيل اضافة طالب جديد	
button	load_btn	تحميل البيانات	
button	insert_btn	اضافة طالب الى قاعدة البيانات	Enabled=False
button	delete_btn	حذف طالب من قاعدة البيانات	
button	update_btn	تعديل المعلومات	
button	dosrch_btn	البحث عن طالب معين	
button	next_btn	السجل اللاحق	
button	prev_btn	السجل السابق	
button	first_btn	السجل الاول	
button	last_btn	السجل الاخير	
button	show_btn	عرض البيانات في الـDgridview	
button	nextpage_btn	عرض الصفحة التالية	
button	prevpage_btn	عرض الصفحة السابقة	
button	current_btn	طباعة السجل الحالي	
button	selected_btn	طباعة السجلات المختارة من الـDgridview	
button	all_btn	طباعة كل السجلات الموجودة في قاعدة البيانات	
Datagridview	DGV	لعرض البيانات	

ثالثا : من Project ثم Add New Item تظهر لك الشكل الآتي:

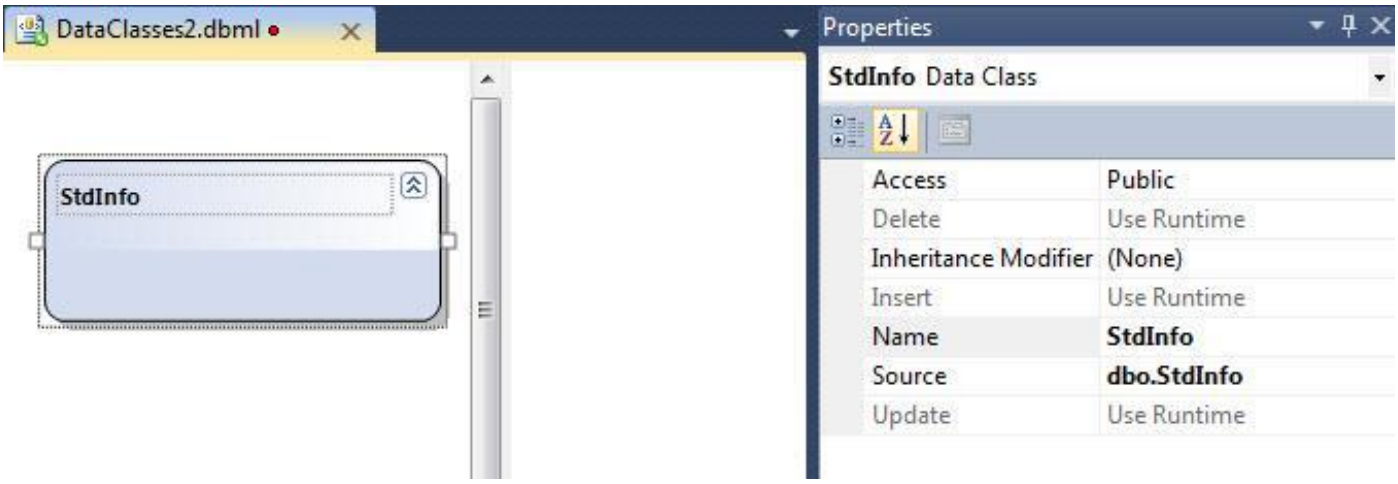


افعل كما في الصورة أي تختار (LINQ to SQL Classes) هذا الـ Class وسيلة لربط قاعدة بيانات SQL بالبرنامج بتقنية الLINQ ومن ثم تضغط على Add فتظهر الشكل الآتي:



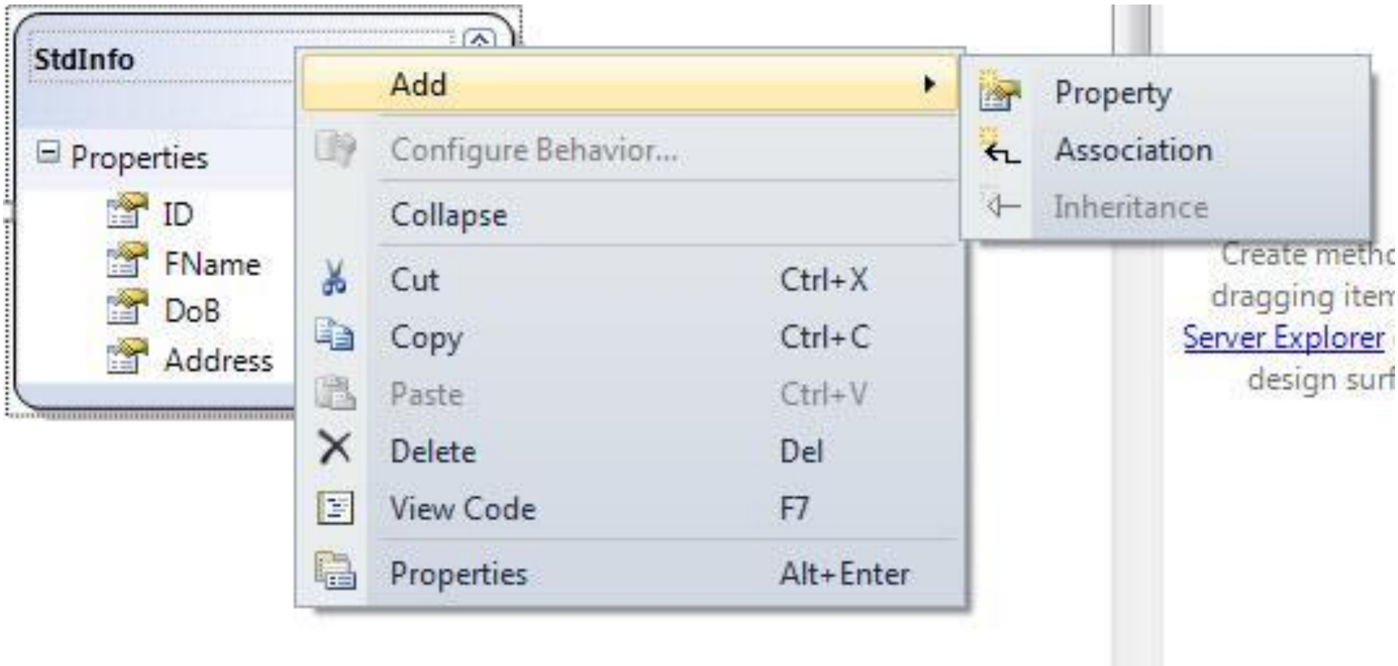
من الخصائص غير الاسم الى **Std_DBase**

وبعد ذلك من الـ **ToolBox** التي على اليسار اسحب أداة **Class** لاحظ الشكل الآتي:

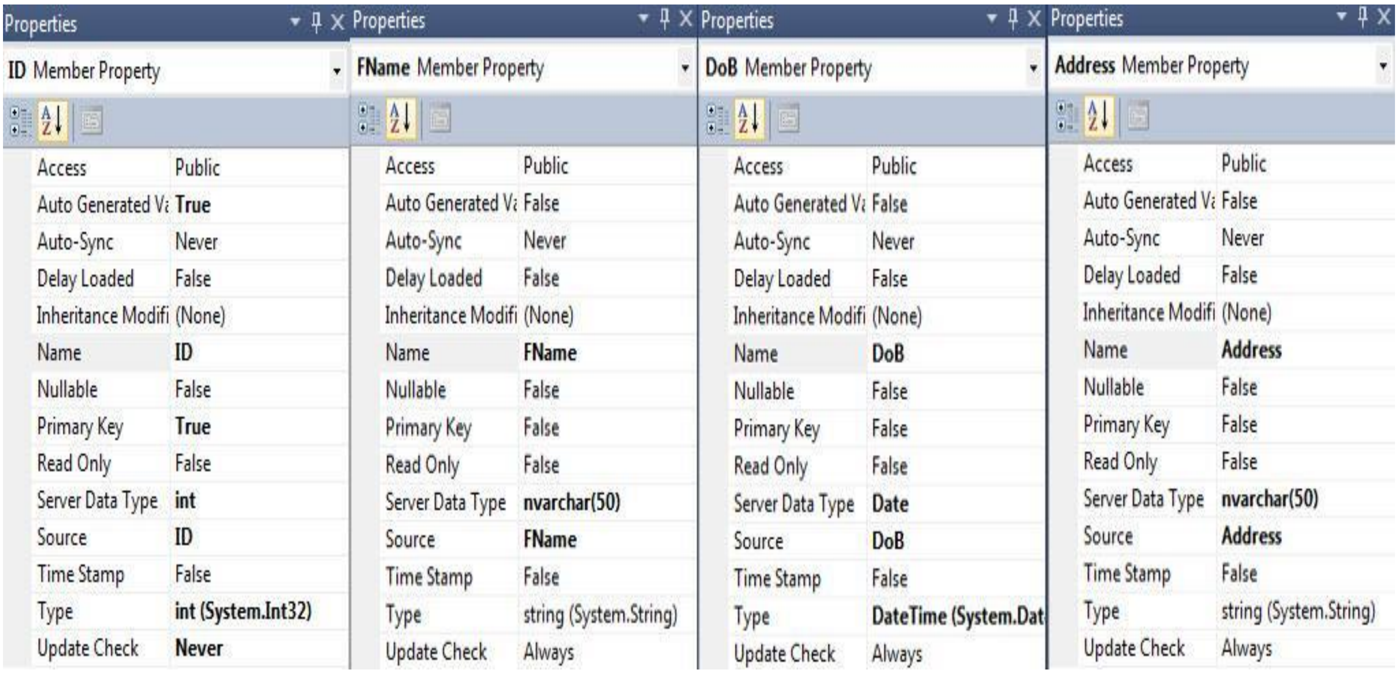


غير اسم الـ **(Class)** والتي يعادل جدول في قاعدة البيانات الى **StdInfo** او أي اسم تختاره ,وفي حقل الـ **Source** اكتب اسم الجدول في قاعدة البيانات وهذا مهم جدا ,لاحظ الشكل اعلاه.

اضيف اربعة خصائص **(Properties)** (حقول) الى الـ **Class** بهذا الشكل:



وقم بضبط خصائص الحقول كما مبين في الشكل أدناه:



ملاحظة : اجعل اسماء الحقول تتطابق مع اسماء الحقول في قاعدة البيانات لسهولة التعامل,لكن تستطيع تغييرها كما تريدها انت.

اهم الملاحظات:

1- جعل خاصية ال **Auto Generated Value** لحقل الـ **ID** يساوي **True** لان الحقل الذي يقابله في قاعدة البيانات ترقيم تلقائي.
2- جعل خاصية الـ **Primary Key** للـ **ID** يساوي**True** .
3- أهم شيء هو تحديد الـ **Source** لكل الحقول ومعناه اسم الحقل الذي يقابله في قاعدة البيانات.
4- تحديد الـ **Server Data Type** كما في الجدول في قاعدة البيانات (**int, varchar, double , ……….**)
5- تحديد الـ **Type** للحقول التي أنشأتها في الـ **Class** بما يطابق نوع الحقول في الجدول في قاعدة البيانات.

رابعا : اضف نموذج الى المشروع لعرض التقرير وسمه **Report Form** ومن ثم اضف اليه اداة **Report Viewer** بعد ذلك من **Project** ثم **Add New Item** اضف تقرير فارغ الى المشروع (لمعرفة المزيد عن تصميم التقارير راجع هذا الرابط: (كيفية تمرير البيانات الى **Microsoft Report** عن طريق البارامترات)

اضيف اربع بارامترات الى التقرير بأسماء, (**ID, Fname, Dob, Address**) ايضا يمكنك مراجعة الرابط اعلاه لمعرفة المزيد. (وأنصح بشدة بذلك).

صمم التقرير بهذا الشكل:



حيث (معلومات الطلبة والعناوين والخطوط الافقية في ال (Page header) والوقت والترقيم في ال Page footer لكي تتكرر في كل الصفحات في حال كون التقرير يتكون اكثر من صفحة.
واضف الباراميترات التي انشأتها الى التقرير كما في الشكل اعلاه.

ملاحظة : يمكن اضافة الـ Page header and footer من شريط الـReport Tool bar

ملاحظة : الـ Page Number والـ Execution Time تستطيع اضافتها من نافذة ال Report Data من الـ -Built-in Fields.

الآن نأتي الى كتابة الاكواد ,وستكون بلغة C#.

اولا: اضف مجالات الاسماء التالية:

C#
```
using System.Collections;
using System.Data.SqlClient;
using Microsoft.Reporting.WinForms;
using System.IO;
```

ثانيا : المتغيرات العامة:

C#

```
// نص الاتصال بقاعدة البيانات
private string constr = @"Data Source=.\SQLEXPRESS;Initial
Catalog=std;Integrated Security=True;Connect Timeout=30;User
Instance=false";

// linqانشاء instance ال تقنية مع تتعامل التي البيانات قاعدة من
private Std_DBase dbo;

// انشاء قائمة من نوع Students لتضم بيانات الطلاب اثناء تحميلها من الجدول في قاعدة البيانات
private List<Students> student;

private int i
        , pos = 0 // موقع السجل (السجلات بين للتنقل)
        , start = 0  // بداية الصفحة من السجل الاول
        , end = 5; // نهاية الصفحة ,هنا حددنا 5 سجلات فقط للصفحة اثناء عرض السجلات في
ال (DataGrudView)

private ReportDataSource rep_source; // لتحديد مصدر البيانات للتقرير
private List<ReportParameter> rep_param; // قائمة من الباراميترات التي ستمرر الى التقرير
```

ثالثا : اضف Class الى المشروع باسم Students لغرض تصفح المعلومات:

C#

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Linq_to_Sql
{
    public class Students
    {
        private int id;
        private string fname, address;
        private DateTime Dob;// (Date of Birth) تاريخ الولادة
```

```
        public int ID
        {
            set { id = value; }
            get { return id; }
        }
        public string Fname
        {
            set { fname = value; }
            get { return fname; }
        }
        public DateTime DoB
        {
            set { Dob = value; }
            get { return Dob; }
        }
        public string Address
        {
            set { address = value; }
            get { return address; }
        }
    }//Class End
}//Namespace End
```

تفعيل اضافة سجل جديد:

C#

```
        private void new_btn_Click(object sender, EventArgs e)
        {
            Enabled_btns(true);
            id_txt.Text = name_txt.Text = dob_txt.Text = addr_txt.Text
= "";
        }
```

دالة التفعيل:

C#

```
        private void Enabled_btns(bool b)
        {
             name_txt.Enabled = dob_txt.Enabled = insert_btn.Enabled =
addr_txt.Enabled = b;
        }
```

الغاء التفعيل:

C#

```
        private void cancel_btn_Click(object sender, EventArgs e)
        {
            Enabled_btns(false);
        }
```

دالة ملئ الحقول:

يتم استخدام كائن الـ**Students** لغرض تصفح البيانات.

C#

```
        private void Fill(int pos)
        {
            Enabled_btns(true);
            insert_btn.Enabled = false;
            id_txt.Text = student[pos].ID.ToString();
            name_txt.Text = student[pos].Fname;
            dob_txt.Text = student[pos].DoB.ToString();
            addr_txt.Text = student[pos].Address;
        }
```

ملاحظة: الكائن StdInfo هو نفس الكائن الذي أنشأتها داخل كائن الـLINQ (DBML).

تحديد نص الاتصال لقاعدة البيانات في حدث الـ Load للـForm

C#

```
private void Form1_Load(object sender, EventArgs e)
{
      // linq تحديد نص الاتصال لقاعدة البيانات التي تتعامل مع تقنية
      dbo = new Std_Dbase(constr);
}
```

<u>شرح الكود:</u>

في هذا الحدث يتم اسناد نص الاتصال بقاعدة البيانات الى Std_Dbase لانشاء الاتصال بقاعدة البيانات.

<u>1- اضافة البيانات:</u>

C#

```
        private void insert_btn_Click(object sender, EventArgs e)
        {
            //Insert Data

            StdInfo std1 = new StdInfo();

            std1.Fname = name_txt.Text;
            std1.DoB = DateTime.Parse (dob_txt.Text);
            std1.Address = addr_txt.Text;

            dbo.StdInfos.InsertOnSubmit(std1);
            dbo.SubmitChanges();

            Enabled_btns(false);
            pos = dbo.StdInfos.Count() - 1;
            load_btn_Click(null, null);
        }
```

<u>شرح الكود :</u>

اولا يتم تعريف متغير من الكائن StdInfo ومن ثم يتم اسناد القيم الى خصائص هذا الكائن وأخيرا يمرر الكائن الى دالة InsertOnSubmit ليتم اضافة القيم الى الجدول في حال تم تأكيد ذلك عن طريق هذه الدالة SubmitChanges لتأكيد التغييرات على الجدول. (جرب ان تحذف الدالة SubmitChanges ثم لاحظ ماذا يحصل).

2- حذف البيانات:

C#

```
        private void delete_btn_Click(object sender, EventArgs e)
        {
            //Delete Data

            if (dbo.StdInfos.Count() != 0)
            {
                StdInfo del =
                  (from StdInfo s in dbo.StdInfos
                   where s.ID == int.Parse(id_txt.Text)
                   select s)
                   .Single();

                dbo.StdInfos.DeleteOnSubmit(del);
                dbo.SubmitChanges();
                pos = dbo.StdInfos.Count() - 1;
                if (pos < 0)
                    pos = 0;
                load_btn_Click(null, null);
            }
        }
```

شرح الكود:

في البداية يتم التحقق من عدم خلو الجدول من سجلات, ومن ثم نستخدم تقنية LINQ لاختيار السجل المساوي لحقل الـID ليتم حذفه.
لاحظ الفرق بين الاستعلام العادي وهذا الاستعلام الخاص بتقنية LINQ المستخدم لحذف السجلات:

```
Delete From table Where Id = @Id
```

الاستعلام أعلاه يتم استخدامه لحذف سجل حسب Id معين من جدول في قاعدة بيانات ما, وتستخدم ايضا تقنية ADO.NET.

الان لاحظ الاستعلام الخاص بتقنية LINQ لحذف سجل معين

```
StdInfo del = (from StdInfo s in dbo.StdInfos
               where s.ID == int.Parse(id_txt.Text)
               select s)
               .Single();
```

في نهاية الاستعلام تم استخدام الدالة Single, هذه الدالة ترجع سجل واحد فقط من نوع الفئة StdInfo وسيتم حدوث استثناء في حال وجود أكثر من سجل يحمل نفس الـ ID.

ويمكن استخدام هذا الكود بدلا من الكود أعلاه وذلك باستخدام تعابير لامبدا (سنستخدم ذلك في تعديل البيانات).

```
StdInfo del = dbo.GetTable<StdInfo>().Where(c => (c.ID  ==
int.Parse(id_txt.Text))).SingleOrDefault<StdInfo>();
// او يمكن استعمال هذه الطريقة وهي تعتبر اسهل الطرق
StdInfo del = dbo.StdInfos.Single(c => c.ID == int.Parse(id_txt.Text));
```

3- تعديل البيانات:

C#

```
        private void update_btn_Click(object sender, EventArgs e)
        {
            //Update Data

            int id = int.Parse(id_txt.Text);

            StdInfo updat = dbo.GetTable<StdInfo>().Where(c => (c.ID ==
id)).Single();
            updat.FName = name_txt.Text;
            updat.DoB = DateTime.Parse(dob_txt.Text);
            updat.Address = addr_txt.Text;

            dbo.SubmitChanges();
            load_btn_Click(null, null);
        }
```

شرح الكود:

التعديل يتم ايضا اعتمادا على الـID الذي هو PrimaryKey, كما بينا سيتم استخدام تعابير لامبدا مع تقنية LINQ في التعديل حسب الـID, بعد اسناد الحقول الى الخصائص يتم استدعاء دالة تأكيد التغييرات على الجدول.

جلب البيانات من قاعدة البيانات:

C#

```
        private void load_btn_Click(object sender, EventArgs e)
        {
            //Load
            if (dbo.StdInfos.Count() > 0)
            {
                var ss =
                    from StdInfo s in dbo.StdInfos
                    select s;

                student = new List<Students>();

                foreach (var std in ss)
                {
                    student.Add(new Students
                    {
                        ID = std.ID,
                        Fname = std.FName,
                        DoB = std.DoB,
                        Address = std.Address
                    });
                    //for autocomplete during searching
                    srch_txt.AutoCompleteCustomSource.Add(std.FName);
                }
                Fill(pos);
            }
            else
            {
                MessageBox.Show("There is no record in table");
            }
        }
```

يتم جلب كافة البيانات عن طريق الاستعلام التالي:

```
var ss = from StdInfo s in dbo.StdInfos select s;
```

ملاحظة: من الممكن كتابة نوع المتغير في صيغة الاستعلام كما في الاستعلام اعلاه, حيث تم تعريف المتغير من نوع StdInfo.

وبعد ذلك يتم المرور على كل السجلات واضافتها الى القائمة التي هي من نوع الكائن Students في هذا الكود:

```
foreach (var std in ss)
                {
                    student.Add(new Students
                    {
                        ID = std.ID,
                        Fname = std.FName,
                        DoB = std.DoB,
                        Address = std.Address
                    });

                    //for autocomplete during searching
                    srch_txt.AutoCompleteCustomSource.Add(std.FName);
                }
```

في آخر سطر يتم اضافة الاسم الى مصدر بيانات التكملة التلقائية للكلمات الخاص بحقل البحث.

عرض اول سجل:

C#

```
        private void first_btn_Click(object sender, EventArgs e)
        {
            Fill(pos = 0);
        }
```

عرض آخر سجل:

C#

```
        private void last_btn_Click(object sender, EventArgs e)
        {
            Fill(pos = student.Count - 1);
        }
```

عرض السجل التالى:

C#

```
        private void next_btn_Click(object sender, EventArgs e)
        {
            if (pos < student.Count - 1)
            {
                pos++;
                Fill(pos);
            }
        }
```

عرض السجل السابق:

C#

```
        private void prev_btn_Click(object sender, EventArgs e)
        {
            if (pos > 0)
            {
                pos--;
                Fill(pos);
            }
        }
```

البحث:

C#

```
private void dosrch_btn_Click(object sender, EventArgs e)
        {

            //Search Data
            load_btn_Click(null, null);
            StdInfo stud =
                (
                 from s in dbo.StdInfos
                 where s.FName == srch_txt.Text
                 select s
                ).Single();
```

```
            id_txt.Text = stud.ID.ToString();
            name_txt.Text = stud.FName;
            addr_txt.Text = stud.Address;
            dob_txt.Text = stud.DoB.ToShortDateString();
        }
```

شرح الكود:

تتم عملية البحث عن طريق الاسم عبر هذا الكود:

```
StdInfo stud =
               (
                from s in dbo.StdInfos
                where s.FName == srch_txt.Text
                select s
               ).Single();
```

لاحظ عملية المقارنة, تكون ناتج المقارنة إما True أو False لهذا السبب يجب كتابة رمز المساواة مرتين بعكس استعلام SQL حيث يتم كتابة الرمز مرة واحدة فقط, وأثناء الكتابة في حقل البحث ستظهر قائمة منسدلة بالاسماء المطابقة حسب خاصية التكملة التلقائية للكلمات.

عرض السجلات في الـDataGridView:

C#

```
        private void show_btn_Click(object sender, EventArgs e)
        {
            DGV.DataSource = "";

            if (dbo.StdInfos.Count() > 0)
            {
                DGV.AutoSizeColumnsMode =
DataGridViewAutoSizeColumnsMode.AllCells;
                if (dbo.StdInfos.Count() <= 5)
                    end = dbo.StdInfos.Count();
                DGV.DataSource = student.GetRange(start, end);
                DGV.Columns[0].HeaderText = "التسلسل";
                DGV.Columns[1].HeaderText = "الاسم الكامل";
                DGV.Columns[2].HeaderText = "تاريخ الولادة";
                DGV.Columns[3].HeaderText = "العنوان";
```

```
            }
        }
```

شرح الكود:

ملاحظة مهمة: تم الاعتماد على الـPaging في عرض السجلات في الـDataGridView, حيث يتم عرض 5 سجلات فقط من الجدول باستخدام دالة الـGetRange للـList الذي هو من نوع الكائن Students , ويمكن تصفح الصفحات في حال وجود اكثر من 5 سجلات في الجدول.

الدالة GetRange تأخذ مدخلين الاول بداية النسخ والثاني كمية النسخ.

عرض الصفحة التالية في الـDataGridView (5 سجلات في كل صفحة):

C#

```
        private void nextpage_btn_Click(object sender, EventArgs e)
        {
            if (start < student.Count - 5)
            {
                start += end;
                if (start + 5 > student.Count)
                    end = student.Count - start;
                DGV.DataSource = student.GetRange(start, end);
            }
        }
```

شرح الكود:

في البداية احب أن أبين المقصود بالـPaging: التصفح معناه جلب جزء من البيانات من الجدول والتعامل معها والفائدة منها هو تخفيف الحمل على الذاكرة, تخيل لو أن الجدول يحتوي على ألاف السجلات فبالتأكيد أن قراءة هذه السجلات يستغرق وقتا طويلا لذا يلجأ الكثير الى مبدأ التصفح أي قراءة جزء من البيانات وهكذا, هنالك طريقة أخرى لقراءة جزء من البيانات وهي عن طريق الاستعلام باستخدام الكلمة المحجوزة Top(number) كالتالي:

```
Select Top(10) * from table
```

عرض الصفحة السابقة فى الـDataGridView:

C#

```
        private void prevpage_btn_Click(object sender, EventArgs e)
        {
            if (start > 0)
            {
                if (end < 5)
                    end = 5;
                start -= end;
                DGV.DataSource = student.GetRange(start, end);
            }
        }
```

الطباعة والتقرير:

1- طباعة السجل الحالى:

C#

```
        private void current_btn_Click(object sender, EventArgs e)
        {
            ReportForm repf = new ReportForm();
            rep_source = new ReportDataSource("student", student);
            repf.reportViewer1.LocalReport.DataSources.Add(rep_source);

            rep_param = new List<ReportParameter>();
            rep_param.Add(new ReportParameter("ID", id_txt.Text));
            rep_param.Add(new ReportParameter("Fname", name_txt.Text));
            rep_param.Add(new ReportParameter("Dob",
DateTime.Parse(dob_txt.Text).ToShortDateString()));
            rep_param.Add(new ReportParameter("Address",
addr_txt.Text));

            repf.reportViewer1.LocalReport.SetParameters(rep_param);
            repf.reportViewer1.RefreshReport();

            repf.ShowDialog();
        }
```

شرح الكود:

في البداية يجب تحديد مصدر البيانات للتقرير وبعد ذلك تمرير البيانات الى التقرير, وبما أننا نعتمد على الباراميترات يجب قبل تمرير الباراميترات ان نسند البيانات التي ستظهر في التقرير الى هذه الباراميترات وذلك عن طريق اسماء البارميترات المصممة في التقرير ويمكن ان تسند البيانات إما عن طريق اسم البارامتير أو الـ**Index** للبارامتير, ويتم ذلك عن طريق الكائن **ReportParameter** وهذا الكائن يأخذ مدخلين الاول إما اسم البارامتير او الـ**Index** للبارامتير حسب الترتيب والثاني البيان (مفرد بيانات) التي ستسند الى البارامتير, وبما أننا بصدد طباعة بيانات السجل الحالي سنسند القيم الحالية الظاهرة في أدوات الـ**TextBox** الى البارامترات كما هو مبين في الكود أعلاه.
وأخيرا نمرر قائمة البارامترات الى التقرير عن طريق الدالة **SetParameters**.

2- طباعة السجلات المحددة من الـ DataGridViewعن طريق الفأرة:

C#

```
private void selected_btn_Click(object sender, EventArgs e)
        {
            ReportForm repf = new ReportForm();

            DataGridViewSelectedRowCollection rc = DGV.SelectedRows;

            rep_source = new ReportDataSource("student", student);
            repf.reportViewer1.LocalReport.DataSources.Add(rep_source);
            int i = 0;
            string s1 = "", s2 = "", s3 = "", s4 = "";

            for (i = 0; i < rc.Count; i++)
            {
                s1 += rc[i].Cells["ID"].Value.ToString() +
Environment.NewLine + Environment.NewLine;
                s2 += rc[i].Cells["FName"].Value.ToString() +
Environment.NewLine + Environment.NewLine;
                s3 += DateTime.Parse(
rc[i].Cells["DoB"].Value.ToString()).ToShortDateString() +
Environment.NewLine + Environment.NewLine;
                s4 += rc[i].Cells["Address"].Value.ToString() +
Environment.NewLine + Environment.NewLine;
            }

            rep_param = new List<ReportParameter>();
            rep_param.Add(new ReportParameter("ID",s1));
```

```
            rep_param.Add(new ReportParameter("Fname", s2));
            rep_param.Add(new ReportParameter("Dob", s3));
            rep_param.Add(new ReportParameter("Address", s4));

            repf.reportViewer1.LocalReport.SetParameters(rep_param);
            repf.reportViewer1.RefreshReport();

            repf.ShowDialog();
        }
```

شرح الكود:

في هذا الجزء سيقوم المستخدم بطباعة السجلات التي سيحددها من الـDataGridView عن طريق الماوس, كأن يحدد سجل او أكثر.

في البداية وقبل تحديد مصدر البيانات للتقرير يجب احتواء الصفوف المحددة باستخدام هذا الكود:

```
DataGridViewSelectedRowCollection rc = DGV.SelectedRows;
```

وظيفة الكود أعلاه هو وضع الصفوف المحددة في Collection من نوع `DataGridViewSelectedRowCollection`, وبعد ذلك يتم تحديد مصدر البيانات ومن ثم تعريف 4 متغيرات من نوع String كل متغير خاص بحقل واحد من الـDataGridView وأخيرا يتم تمرير هذه المتغيرات التي تضم الحقول المحددة الى التقرير.

3- طباعة كل السجلات:

C#

```
        private void all_btn_Click(object sender, EventArgs e)
        {
            ReportForm repf = new ReportForm();

            rep_source = new ReportDataSource("student", student);
            repf.reportViewer1.LocalReport.DataSources.Add(rep_source);

            int i = 0;
            string s1 = "", s2 = "", s3 = "", s4 = "";

            for (i = 0; i < student.Count; i++)
```

```
            {
                s1 += student[i].ID.ToString() + Environment.NewLine +
Environment.NewLine;
                s2 += student[i].Fname.ToString() + Environment.NewLine
+ Environment.NewLine;
                s3 +=
DateTime.Parse(student[i].DoB.ToString()).ToShortDateString() +
Environment.NewLine + Environment.NewLine;
                s4 += student[i].Address.ToString() +
Environment.NewLine + Environment.NewLine;
            }

            rep_param = new List<ReportParameter>();
            rep_param.Add(new ReportParameter("ID", s1));
            rep_param.Add(new ReportParameter("Fname", s2));
            rep_param.Add(new ReportParameter("Dob", s3));
            rep_param.Add(new ReportParameter("Address", s4));

            repf.reportViewer1.PageCountMode = PageCountMode.Actual;
            repf.reportViewer1.LocalReport.SetParameters(rep_param);
            repf.reportViewer1.RefreshReport();
            //Showing the report form
            repf.ShowDialog();
        }
```

شرح الكود:

أخيرا طباعة كل ما موجود من سجل في الجدول, بينا فيما سبق أنه اثناء تحميل البيانات يتم وضعها في قائمة من نوع Students اذن كل البيانات توجد في هذه القائمة لذا يتم المرور على كافة السجلات ويتم وضعها كما ذكر سابقا في المتغيرات النصية ومن ثم يتم تمريرها الى التقرير.

و أخيرا لا تنسوني و والدي من صالح دعواتكم و إن اخطأت فهو مني و إن اصبت فمن الله

والله ولي التوفيق

البريد الألكتروني: sajad_88m@yahoo.com .